art_root_io - Bug #25263

art memory use for concatenation

11/30/2020 02:37 PM - Andrei Gaponenko

Status:	Closed	**Start date:**	11/30/2020
Priority:	Normal	**Due date:**	
Assignee:	Kyle Knoepfel	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.05.00	**Spent time:**	6.00 hours
Scope:	Internal	**Experiment:**	Mu2e

Description

Hello,

I have a dataset of less than 10,000 files, about 90 kB each. When I
run art to concatenate them into a single file, the process gets
killed after exceeding 11229796maxresident size (~11GB).

The files are both produced and concatenated with art v3_06_03 (using
the HEAD of Mu2e Offline). A configuration file for running the
concatenation is attached.

The memory requirement does not seem to be reasonable for the job.
Please investigate and improve.
Andrei

History

#1 - 11/30/2020 02:38 PM - Andrei Gaponenko

- File cnf.gandr.potToExtMonRoom-QGSP_BERT-cat.9747files.000001_00000001.fcl added

#2 - 11/30/2020 04:11 PM - Kyle Knoepfel

- Assignee set to Kyle Knoepfel

- Status changed from New to Feedback

Is it sufficient to use v09_11_00 of Mu2e Offline?

I'm running a concatenation job now with that version on mu2egpvm01. After processing 1260 input files, the RSS has gradually climbed to about 660 MB. I expect I'll see more dramatic memory growth as the job progresses...?

#3 - 11/30/2020 10:48 PM - Andrei Gaponenko

- File memlog.dat added

- File memory_vs_nfiles.pdf added

Here is an update. I managed to concatenate the whole dataset. The
output file size is only 64 MB. However the resident memory size of
the concatenation process grew to 19.6 GB per a /usr/bin/time
printout. This used
/cvmfs/mu2e.opensciencegrid.org/Offline/v09_11_00/SLF7/prof/Offline/setup.sh

Attached are a plot of resident memory size vs the number of open
files, and a "raw data" file for the plot (which is mostly a text
file). The first column in memlog.dat is time since the beginning of
the observation, in seconds. The second is the number of files opened
by art so far, per the "Opened input file" printout. The rest is the
content of /proc/$PID/statm file at that time; the second value (the
fourth column in memlog.dat) is the resident memory size of the
process in pages.

Memory grows quadratically with the number of files.

Andrei

#4 - 12/01/2020 03:15 PM - Kyle Knoepfel

- % Done changed from 0 to 100

- Status changed from Feedback to Resolved

- Project changed from art to art_root_io

The issue is understood. When opening each new input file, the framework checks that the products to be forwarded to the output file have been properly registered. Instead of just checking for relevant information--i.e. the module label, C++ type, instance label, process name, and processing level (event, subrun, etc.)--the registration process also checks for identical configurations. If the configuration IDs are different, then separate entries are cached in art's product-I/O system, potentially leading to enormous memory footprints, which your use case exposed.

I have adjusted the product-from-input registration system so that only the relevant information is used to assemble the list of products to be written to disk. Whereas I was seeing 2.7 GB max RSS after processing 3400 files (consistent with your findings), with the bug fix the maximum RSS is 400 MB processing the same number of files.

This bug fix (see commit art_root_io:cbadc3) will be included in the next art_root_io version, which is being tagged today.

#5 - 12/01/2020 03:46 PM - Kyle Knoepfel

- Target version set to 1.05.00

#6 - 12/01/2020 04:09 PM - Rob Kutschke

Thanks Kyle. What's the ETA on the next planned release?

Andrei: when do you need this? After Kyle answers, do we need a bug fix release on a faster timescale?

#7 - 12/01/2020 04:33 PM - Kyle Knoepfel

I've posted the new critic release just recently (https://scisoft.fnal.gov/scisoft/bundles/critic/v2_05_00/critic-v2_05_00.html), which includes the new ROOT version with TEve7 and this bug fix.

Assuming Eric can tag a relevant version of artdaq_core, I should be able to start the mu distribution tomorrow.

#8 - 12/01/2020 04:41 PM - Andrei Gaponenko

Wow! That was fast! Thank you, Kyle!
Andrei

#9 - 12/01/2020 10:28 PM - Andrei Gaponenko

Hi Kyle, a question on the problem and fix. Per your comment #4 above the problem was due to per-file configuration storage. However than the memory growth would be linear in the number of files. What we observe is quadratic. (I did not include it in the plot, but the curve fits nicely with a parabola.) How do we reconcile that?
Andrei

#10 - 12/02/2020 09:47 AM - Kyle Knoepfel

- Status changed from Resolved to Closed

#11 - 12/02/2020 10:54 AM - Kyle Knoepfel

- Status changed from Closed to Assigned

Andrei, you are correct that the per-file, product-registry memory behavior would account for linear growth. However, as the product registry accumulates entries, each increase in the size of the registry triggers additional memory allocations to prepare for writing products to disk. And I believe the algorithm for doing that results in linear growth per product-registry entry. Those two behaviors combined would lead to quadratic memory growth. If indeed that is the case, then I suspect a memory leak somewhere in the RootOutput module.

I will investigate further and let you know what I find out. Regardless, you should be able to use the new art_root_io version, as the memory growth is greatly mitigated even if a memory leak (not memory corruption) exists.

#12 - 12/02/2020 11:03 AM - Kyle Knoepfel

The mu distribution is available at https://scisoft.fnal.gov/scisoft/bundles/mu/v3_06_03a/mu-v3_06_03a.html.

#13 - 12/02/2020 12:14 PM - Rob Kutschke

Thanks Kyle. Two things.

We are currently using e20, not e19. Please also make the e20 bundles, prof and debug.

I see that the version and qualifiers of art have not changed. Our standard practice is to "setup -B art version qualifiers" and everything else comes along for the ride. What is the prescription to get this software stack?

#14 - 12/02/2020 12:33 PM - Kyle Knoepfel

Starting the e20 builds now; will let you know when they're done--sorry about that.

Because art does not depend on ROOT, there is no change in art version. To get the new ROOT version plus the bug fix, you'll have to setup art_root_io 1.05.00 instead of 1.04.03, and mu2e's setup script already sets up art_root_io separately from art.

#15 - 12/02/2020 01:24 PM - Kyle Knoepfel

e20 builds have been posted to SciSoft

#16 - 12/03/2020 08:43 AM - Rob Kutschke

Thanks Kyle. I noticed that you also upgraded root to v6_22_2 from v6_20_0 .   Is this built with root7 support or not?

#17 - 12/03/2020 08:50 AM - Kyle Knoepfel

Yes, the SciSoft build of ROOT v6_22_02 includes the TEve7 components. Because TEve7 is just an add-on to the standard ROOT build, we will enable it for future ROOT builds as well. That way nobody has to worry about special version numbers, letters or qualifiers.

#18 - 12/12/2020 08:23 PM - Andrei Gaponenko

- File memory_vs_nfiles-fixed.pdf added

I have re-run the same concatenation job using mu2e Offline build against ART_ROOT_IO_VERSION=v1_05_00.
A memory-vs-nfiles plot is attached. The curve is flat now; I confirm that the issue has been fixed.

Thank you!
Andrei

#19 - 12/14/2020 10:19 AM - Kyle Knoepfel

- Status changed from Assigned to Resolved

#20 - 12/14/2020 04:32 PM - Kyle Knoepfel

- Status changed from Resolved to Closed

Files

cnf.gandr.potToExtMonRoom-QGSP_BERT-cat.9747files.000001_000[illegible].fcl	[illegible] MB	11/30/2020	Andrei Gaponenko
memory_vs_nfiles.pdf	23.9 KB	12/01/2020	Andrei Gaponenko
memlog.dat	174 KB	12/01/2020	Andrei Gaponenko
memory_vs_nfiles-fixed.pdf	14.2 KB	12/13/2020	Andrei Gaponenko